Linguistic Knowledge and Word Sense Disambiguation

Gaustad, Tanja

Chapter 6

Impact of Part-of-Speech Information

After having explored the benefit of using morphological information for WSD of Dutch, we will now proceed to integrate part-of-speech (PoS) information into the WSD system explained in chapter 4. The Dutch Senseval-2 WSD data is not only ambiguous with regard to meaning, but also with regard to PoS. This means that accurate PoS information is important since the WSD system is supposed to do morpho-syntactic as well as semantic disambiguation. First, we investigate which of three different PoS taggers performs best in our system in an application-oriented evaluation. Following the strategy that high quality input is likely to influence the final results of a complex system, we test whether the more accurate taggers also produce better results when integrated into the WSD system. For this purpose, a stand-alone evaluation of the PoS taggers is used first to assess which tagger is the most accurate.

There are two possible outcomes to the experiment of integrating PoS information in our WSD system. Either PoS does not improve the performance of the algorithm because the information added by the PoS tags is already implicitly contained in other features (or because it is irrelevant), or PoS does help disambiguation since PoS disambiguation is part of the initial problem in the case of the Dutch Senseval-2 Data. The results of the WSD task including the PoS information from all three taggers show that including PoS information in the WSD system improves accuracy and that the most accurate PoS tags indeed lead to the best results, thereby verifying our hypothesis.

In the second part of the chapter, not only PoS of the ambiguous word form, but also PoS of the context is added. This allows us to test the disambiguation value of PoS information on a greater scale and in a novel way. The

results show that accurate PoS information is beneficial for WSD and that including the PoS of the ambiguous word itself as well as PoS of the context increases disambiguation accuracy over the system presented in chapter 4.

6.1 Application-Oriented Evaluation of Three PoS Taggers

Certain NLP tools are typically used as a sub-component or a pre-processor in a more complex system, rather than as a complete application in their own right. A typical example of such tools are PoS taggers. What is usually not taken into account is the fact that the quality (in terms of accuracy) of each sub-part of a complex system is likely to influence the final results considerably. Lately, standardized evaluation of NLP resources has gained more importance in the field of computational linguistics (e.g. CLEF workshops in information retrieval, Parseval, SENSEVAL), but a tendency towards more application-oriented evaluation is only beginning.

In the first part of the chapter, we will proceed to an application-oriented comparison of three PoS taggers in a word sense disambiguation (WSD) system. We will evaluate to what extent differences in stand-alone PoS accuracy influence the results obtained in the complex WSD system using the acquired PoS information. Since the Dutch data we use is not only ambiguous with regard to meaning but also with regard to PoS, accurate PoS information is potentially very important to achieve high disambiguation accuracy.

We will start with a detailed description and comparison of the three PoS taggers including a stand-alone evaluation in order to compare their performance independently of the application to the WSD task. Then follows a short description of how the output of the different PoS taggers is incorporated into the maximum entropy WSD system for Dutch explained earlier. Next, the application-dependent results of the three PoS taggers will be presented and discussed.

6.2 Comparison of PoS Taggers

The PoS taggers we compare are:

- a Hidden Markov Model tagger (section 6.2.1),
- a Memory-Based tagger (section 6.2.2),
- a transformation-based tagger (section 6.2.3).

We chose these three taggers because they were readily available, could easily be trained for Dutch without major changes in the architecture, and represent distinct, widely used types of existing PoS taggers.

All three taggers were trained on the Dutch Eindhoven corpus (uit den Boogaart, 1975) using the WOTAN tag set (Berghmans, 1994). The original WOTAN tag set, consisting of 233 tags, was too detailed for our purpose. Instead, we used the limited WOTAN tag set of 48 PoS tags developed by Drenth (1997) for training and testing in the stand-alone comparison of the three PoS taggers.

In the context of our WSD system, however, we are chiefly interested in the main PoS categories only. Therefore, we discarded all additional information from the assigned PoS tags in the WSD corpus. This resulted in 12 different tags being kept: Adj (adjective), Adv (adverb), Art (article), Conj (conjunction), Int (interjection), Misc (miscellaneous), N (noun), Num (numeral), Prep (preposition), Pron (pronoun), Punc (punctuation), and V (verb).[1]

For the stand-alone results, 80% of the annotated data was actually used for training, 10% for tuning (setting of parameters, etc.) and the accuracy was computed on the remaining 10%. Note that the results of the stand-alone comparison solely serve to illustrate the difference in performance observed independently of an application in order to be able to assess the added value of a more accurate PoS tagger in the WSD application.

6.2.1 Hidden Markov Model PoS Tagger

The first PoS tagger we used is the trigram Hidden Markov Model (HMM) tagger (Prins and van Noord, 2004) developed in the context of Alpino, a natural language understanding system for Dutch (Bouma et al., 2001; van der Beek et al., 2002).[2]

In this standard trigram HMM, each state corresponds to the previous two PoS tags and the probabilities are directly estimated from the labeled training corpus (Manning and Schütze, 1999). There are two types of probabilities relevant in this model, the probability of a tag given the preceding two tags $P(t_i|t_{i-2}t_{i-1})$ as well as the probability of a word given its tag $P(w_i|t_i)$.

These probabilities are computed for each tag individually. Training the HMM with the forward-backward algorithm, we can calculate $P(t_i = t)$ for all potential tags:

[1] See table 6.2 for a distribution of the main PoS tag categories in the WSD data and the Eindhoven corpus.

[2] See `http://www.let.rug.nl/~vannoord/alp` and chapter 7 for more information on Alpino.

$$P(t_i = t) = \alpha_i(t)\beta_i(t)$$

where $\alpha_i(t)$ is the total (summed) probability of all paths that end at tag t at position i, and $\beta_i(t)$ is the total probability of all paths starting at tag t in position i continuing to the end. Comparing all the values for $P(t_i = t)$, unlikely tags are removed.

Smoothing of the trigram probabilities is achieved through a variant of linear interpolation (Collins, 1999) where lower order (unigram) models are also taken into account and weights are assigned to each of the models to capture their relative importance.

Since the tagger's lexicon has been created from the training data, the test data very likely contains unknown words which means that no initial set of possible tags can be assigned to these words. Two different strategies have been incorporated in the HMM tagger used here. First, a heuristic rule for recognizing names has been added which assigns an N tag to all capitalized words.[3] Second, a set of automata (also created on the basis of the training data) is used to find possible tags based on the suffixes of unknown words (Daciuk, 2000).

6.2.2 Memory-Based PoS Tagger

The second tagger we have used in the experiments reported here is the Memory-Based Tagger (MBT) (Daelemans et al., 2002a).[4] It is a PoS tagger based on Memory-Based Learning, an extension of the k-Nearest-Neighbor approach, which has proved to be successful for a number of languages and NLP applications (Zavrel and Daelemans, 1999; Veenstra et al., 2000; Hoste et al., 2002a).

MBT consists of two components: a memory-based learning component and a performance component for similarity-based classification. During classification, the similarity between a previously unseen test example and the examples in memory is computed using a similarity metric. The category of the test example is then extrapolated based on the most similar example(s).

Given an annotated corpus, three data structures are automatically extracted: a lexicon, a case base[5] for known words, and a case base for unknown words. During tagging, each word is looked up in the lexicon and, if it is found, its lexical representation is retrieved and its context determined. The resulting pattern is disambiguated using extrapolation from the

[3] Words in sentence initial position are decapitalized beforehand.

[4] Freely available for research purposes at `http://ilk.uvt.nl/software.html`.

[5] A case base is a collection of "cases" or prior observations.

nearest neighbors in the known words case base. If a word is not present in the lexicon, its lexical representation is computed on the basis of its form, its context is determined, and the resulting pattern is disambiguated using extrapolation from nearest neighbors in the unknown words case base. In both cases, the output is a best guess of the category for the word in its current context.

For the known words, the preceding two tags and words as well as the ambiguous tag and word to the right of the current position have been used to construct the known words case base. Classification was achieved using the IGTREE algorithm with one nearest neighbor. For unknown words, the preceding tag, the ambiguous tag to the right, as well as the first and the last three letters of the ambiguous word itself were taken into account to construct the unknown words case base. For classification, the IB1 algorithm with 9 nearest neighbors was used. In both cases GainRatio feature weighting was applied. For details on the different algorithms, IGTREE, IB1, and GainRatio, see Daelemans et al. (2002b).

6.2.3 Transformation-Based PoS Tagger

As the third member of the comparison, we used a Brill-style transformation-based tagger (TBL) (Brill, 1995) for Dutch (Drenth, 1997). The main components of a transformation-based tagger are a specification of admissible transformations and a learning algorithm. Interdependencies between words and tags are modeled by starting out with an imperfect tagging which is gradually transformed into one with fewer errors. This is achieved by selecting and sequencing transformation rules using the learning algorithm.

In an initial step, each word is assigned a tag independent of context. A known word is assigned its most likely tag determined by a maximum likelihood estimation from the training corpus. An unknown word, on the other hand, is assigned a tag based on lexical rules learned during training. All unknown words are initially tagged N. The application of lexical rules modifies the tag (where necessary) based on the local properties of the unknown word, such as its suffix.

After each word has received an initial tag, contextual rules are applied changing the initial PoS tag (where necessary) based on the context of the word to be tagged (templates taken from (Brill, 1995)). The best contextual transformation rules and their order of application are selected by the learning algorithm during training.

The present implementation of the TBL PoS tagger for Dutch uses around 250 lexical rules and 300 contextual rules.

PoS Tagger	Accuracy
TBL	94.20
HMM	95.93
MBT	**96.21**

Table 6.1: Stand-alone results (in %) for the three PoS taggers on 10% of the Eindhoven corpus data.

6.2.4 Stand-Alone Results for the PoS Taggers

As we have mentioned earlier, the stand-alone results for the PoS taggers were computed using 80% of the Eindhoven Corpus (containing a total of 760,000 words) for training and 10% for tuning. The accuracy shown in table 6.1 was computed on the remaining 10% of the corpus.

We can clearly see that the MBT tagger is performing best, followed by the HMM tagger, the least accurate tagger being the TBL tagger.[6] If the hypothesis that more accurate input to complex systems will produce more accurate results is correct, then these stand-alone results raise the expectation that when applying all three taggers in our WSD system—with all other settings being equal—accuracy should be highest when the MBT tagger was used to tag the data. Performance is expected to decrease with the use of the HMM tagger and to be lowest for the TBL tagger.

This expectation might be invalidated by the (possible) corpus dependency of the three PoS taggers: the capacity to generalize from the training corpus to the corpus to be tagged might be bigger in one tagger than in another, which means that the results obtained in the complex system can diverge from the expectation raised by the stand-alone results. Also, it may be that a tagger is more accurate than another but mainly on distinctions that are unimportant for our WSD application.

6.3 Integrating PoS Information

The WSD system used in these experiments is the same supervised corpus-based statistical classification algorithm described in section 4.2.1. This system explores the intuition that (high quality) linguistic information is beneficial for WSD. PoS is definitely one of the more accessible sources of linguistic knowledge. The hypothesis behind comparing various PoS taggers

[6] All results differ significantly applying the paired sign test with a confidence level of 95%.

PoS	TBL	HMM	MBT	Train. Corpus
N	19.46%	17.08%	17.37%	20.35%
Punc	16.87%	17.17%	17.17%	12.69%
V	15.04%	16.62%	16.66%	15.13%
Pron	11.83%	11.88%	11.83%	9.82%
Adv	9.58%	9.62%	9.53%	8.19%
Art	8.08%	7.96%	7.95%	9.39%
Prep	6.98%	7.26%	7.01%	10.54%
Conj	5.72%	5.74%	5.77%	5.18%
Adj	5.38%	5.63%	5.65%	6.53%
Num	0.74%	0.61%	0.63%	1.78%
Int	0.32%	0.47%	0.39%	0.18%
Misc	0.003%	0.04%	0.04%	0.22%

Table 6.2: Frequencies of PoS tags assigned by each PoS tagger in the Dutch SENSEVAL-2 WSD data and distribution of PoS in the Eindhoven training corpus.

in this application is that the quality of the PoS tags assigned to the data can significantly influence the accuracy obtained by our WSD system.

In contrast to the English WSD data, the Dutch SENSEVAL-2 WSD data is ambiguous with regard to PoS. This means that accurate PoS information is even more important since the WSD system is supposed to do morpho-syntactic as well as semantic disambiguation.

For the two *basic classifiers* based on ambiguous word forms, the feature set contains the corresponding lemma as well as a context of three words to the left and to the right of the ambiguous word. The context can either be composed of word forms or lemmas. For the *classifiers including PoS tags*, we in addition include the PoS tags of the ambiguous word from the various PoS taggers.

6.4 Results and Evaluation

Before we turn to the actual results of using the different PoS taggers in our WSD system for Dutch, let us first compare the differences regarding the assigned PoS tags. Table 6.2 shows the distribution of the different PoS tags in the WSD data depending on the PoS tagger used, as well as the distribution of the PoS tags in the training corpus.

A major difference between the distribution of PoS tags is that both the

	Accuracy		
baseline training data	75.64		
lemma, context words	83.32		
lemma, context lemmas	83.43		
	TBL	HMM	MBT
lemma, pos, context words	83.53	83.63‡	**83.72**†‡
lemma, pos, context lemmas	83.67	83.76‡	**83.82**†‡

Table 6.3: Results (in %) using leave-one-out on training data with a Gaussian prior of 1000, integrating the output from different PoS taggers; † denotes a significant improvement over the results with the HMM tagger, ‡ denotes a significant improvement over the results with the TBL tagger.

HMM and MBT tagger assign more V tags, whereas the TBL tagger assigns more N tags. The preference for N tags in the TBL tagger can be explained by the fact that all unknown words initially get tagged N. Also, in Dutch, verbal infinitives have the same morphological suffix as plural nouns (*-en*). INT and Misc differ with all three taggers, but we could not detect any obvious reason for this. As we can see from table 6.2, there are bigger differences between the TBL tagger and the other two, whereas the differences between the HMM and the MBT tagger are less noticeable.

The results in table 6.3 show the average accuracy on our training data using leave-one-out as a test method with word forms as basis.[7] As the table of results shows, the WSD system performs well. The basic classifiers containing a minimum of information already achieves significantly better results than the frequency baseline. Furthermore, adding (machine-generated) PoS as extra linguistic information—next to the lemma and the context already included in the basic classifiers—increases results over the accuracy achieved with a basic classifier.

This supports the underlying hypothesis behind the WSD system that more linguistic information is beneficial for WSD. Since the WSD data needs to be disambiguated morpho-syntactically as well as with regard to lexical semantic ambiguity, it is not surprising that adding PoS information achieves better results than only using the lemma and context.

Comparing the performance among the different PoS taggers, we can see quite clearly that our expectations are confirmed: the MBT tagger, which did best in the stand-alone evaluation, is also working best in the WSD system.

[7]The results in this table differ slightly from the results presented in Gaustad (2003) since we added smoothing with Gaussian priors and did not use a bag of words approach for the context in the experiments reported here.

	w/o PoS	with PoS	
PoS ambiguous: 204 lemma, context lemmas	84.63	85.36†	(+0.73)
non-PoS ambiguous: 749 lemma, context lemmas	81.80	81.74	(−0.06)

Table 6.4: Comparison of accuracy with more than one PoS tag assigned by the PoS tagger; † denotes a significant improvement over not including PoS.

This is the case for classifiers including context as word forms or as lemmas.[8] We are conscious of the fact that the differences in accuracy are rather small, but since the models are very similar no big differences were expected. This means that our hypothesis that highly accurate input influences the results of a complex system is verified: the most accurate PoS tags also produce the most accurate results when integrated into our WSD system.

In order to get a better picture of the effect of adding machine-generated PoS information to our feature model, we proceeded to a more detailed evaluation. In particular, we analyzed whether adding PoS information of the actual ambiguous word form improves the performance on PoS ambiguous words. There are two ways in which a word can be PoS ambiguous: it can either be (incorrectly) assigned more than one PoS tag by the tagger (PoS ambiguity generated through the tagger) or it can really be PoS ambiguous. Since the second case is harder to verify, we started with the first and more important one.

Extracting all word forms that are assigned more than one PoS tag by the MBT PoS tagger, we retain 204 word forms of the total 953 (21.4%). Comparing the accuracy on these words we see clearly that adding PoS information helps disambiguation (see table 6.4): An error rate reduction of 4% is achieved when PoS information of the ambiguous word is included.

The accuracy of the remaining non-PoS ambiguous word forms decreases slightly due to the fact that some noise is added through the PoS information, but the differences are rather small. We can also see these results in a positive light: it means that if we add noisy features to the model, it does not have a big influence on the accuracy. In other words, the model is robust and does not assign high weights to useless features. We can conclude from this

[8]Applying the paired sign test with a confidence level of 95%, all results using MBT PoS tags were found to be statistically significantly better than results with other PoS tags (and than the basic classifiers). The same is true of the HMM PoS tagger with regard to the TBL tagger and the basic classifiers, and the TBL tagger with regard to the basic classifiers (see section 4.5 on the paired sign test).

	Accuracy		
baseline training data	75.64		
lemma, context words	83.32		
lemma, context lemmas	83.43		
	TBL	HMM	MBT
lemma, pos, context words	83.53	83.63‡	83.72†‡
lemma, pos, context lemmas	83.67	83.76‡	83.82†‡
lemma, context words, pos in context	84.16	84.21	84.22
lemma, context lemmas, pos in context	84.20	84.28	84.29
lemma, pos, context words, pos in context	84.21	84.21	**84.31**†‡
lemma, pos, context lemmas, pos in context	84.23	84.34‡	**84.36**‡

Table 6.5: Results (in %) using leave-one-out on training data with a Gaussian prior of 1000, including PoS of the ambiguous word form and PoS of context; † denotes a significant improvement over the results with the HMM tagger, ‡ denotes a significant improvement over the results with the TBL tagger.

more detailed analysis that PoS of the ambiguous word is definitely a useful feature in the case of the Dutch SENSEVAL-2 data.

6.5 PoS Information in Context

As we have seen, adding PoS information of the ambiguous word significantly increases the accuracy achieved by making morpho-syntactic ambiguity distinctions easier. In order to test the use of PoS on a greater scale, we also examined the effect of including PoS for all context words or lemmas (see table 6.5).

Using the same basic model as in section 6.4, the PoS of the context was added. One model contained the lemma of the ambiguous word, the context and its PoS. This allowed us to assess the added value of using PoS for the context in comparison to only using the context alone. The other model included the PoS of the ambiguous word along with the features contained in the first model to test the complementarity of the features.

As can be seen in table 6.5, adding the PoS of the context performs better than all the models tested so far, also the one including the PoS of the ambiguous word (repeated in table 6.5 to ease comparison). It seems to be the case that adding (automatically generated) PoS of the context helps disambiguation by presenting more general information than context words

or lemmas and therefore countering possible data sparseness. Also, PoS of the context can be seen as a very raw form of subcategorization information, indisputably a useful feature especially for verbs.

Combining both the PoS of the ambiguous word and the PoS of the context also has a positive effect on the accuracy of our classifiers, although the increase in accuracy is not as big as with adding PoS of the context. This seems to suggest that combining features improves results.

Comparing the results achieved with the three different PoS taggers, we can only conclude that the TBL tagger performs significantly worse.[9] The MBT tagger and the HMM tagger do not perform significantly differently, but for both we can see a significant increase in performance when adding PoS information of the context only or in combination with the PoS of the ambiguous word.

The results presented in this chapter lead to the conclusion that including PoS information for both the ambiguous word and the context in our feature model significantly improves the performance of the maximum entropy WSD system for Dutch over the model which does not. We mentioned two possible outcomes for the experiment where the PoS of the ambiguous word is added to the feature set. On the one hand, performance could have stayed the same because no new information was added to the model since the PoS information was already implicitly contained in other features, or, on the other hand, including PoS could lead to higher accuracy by adding new knowledge. As we have shown, the latter is the case. We therefore conclude that explicitly encoding information that helps disambiguation is better than relying on maximum entropy to filter out knowledge that is maybe implicitly present in the features used. The bias of maximum entropy does not seem to capture this information otherwise.

[9]Except for the model including the lemma, PoS of the ambiguous word, as well as context words where all three taggers do not perform significantly differently.